Docket # 4283
INV.: Dieter ANSEL

Contactless data transmission system

BACKGROUND OF THE INVENTION

The invention relates to a contactless data transmission system in accordance with the preamble of Patent Claim 1.

For the transmission of encoded electronic data, data transmission systems are commonly used comprising a battery operated (IR/HF) remote control as transmitter module and a suitable receiver module. Furthermore, data transmission systems consisting of a transponder and a reader are also used in which initially by means of an inductive coupling power transmission takes place between reader and transponder and subsequently data transmission between transponder (transmitter) and reader (receiver). Data transmission systems of this kind are available on the market, especially in the field of motor vehicles, as a combination of electronic key (transponder) and electronic lock (reader) for the purpose of operating lock systems and antitheft devices in the form of immobilizers.

Data transmission in the RF range (typically 100 kHz to 450 kHz) between the transponder, which essentially consists of an integrated circuit (IC) and a coil, and the reader can take place in several ways.

Either unidirectionally by means of a fixed-code transponder which transfers as password each time readout takes place (each time data is transmitted) a fixed code stored in a programmable read-only memory (PROM) of the IC or bidirectionally by means of a read/write transponder which transfers as password each time readout takes place (each time data is transmitted) a variable code stored in a read/write memory

(EEPROM) of the IC and after successful authorization from the reader receives a new code and enters this in the EEPROM. As an additional measure for enhancing security, not only are ciphering methods adopted but also algorithms are used to verify that the transponder and base station belong together. The following sequence results:

- the base station generates an electromagnetic field;
- this causes the transponder to be activated;
- the transponder sends its identification number to the base station;
- the base station checks the correctness of the identification number and generates a base station random number;
- the base station random number is ciphered in the base station;
- the ciphered base station random number is sent to the transponder where it is deciphered and thereby generates a transponder random number;
- this transponder random number or a value dependent on the transponder random number is sent to the base station where it is checked for its correctness;
- this transponder or base station random number is the input value for an algorithm that includes a variable which exists both in the base station and in the transponder, unique values being assigned to the variable of the algorithm through a secret code thus generating a transponder result in the transponder and a base station result in the base station.
- The transponder result is sent to the base station.
- Transponder result and base station result are compared in the base station.

In such a sequence, identical results from transponder and base station can be obtained only when the secret code, the algorithm, the random number and the cipher in the two components are identical or at least known by the other component.

The security and also the reaction speed of such a data transmission system depends among other factors on the format of the random number, and especially on the number of bits that make up the random number.

It is however **disadvantageous** here that the security and possibly the range and the reaction speed of a contactless data transmission system containing an algorithm for encoding cannot be modified.

SUMMARY OF THE INVENTION

The object of the invention is to provide a contactless data transmission system in which the reaction speed, the range and security can be subsequently modified, in particular depending on the application.

The object of the invention has been **solved** by the features described in Patent Claim 1. The data transmission system here has at least one device with which the various input data formats for the encoding algorithm are set. The device can consist of one or several additional hardware terminal connections or terminal connection assignments and switches or it can consist of one or several additional control signals that determine the input data format.

The **advantages** of the invention are that it is no longer necessary to have different data transmission systems for different applications, instead identical data transmission systems can be used for different applications with different requirements. Also, the properties of such data transmission systems can be set individually for one and the same application.

Advantageous further developments result from the subclaims where one and the same encoding algorithm is used for the various input data formats. Another advantageous further development results from the retention of the secret code irrespective of the input data format.

BRIEF DESCRIPTION OF THE DRAWINGS

The invention will now be described in more detail with reference to two examples of embodiment and figures. These show:

Figure 1: Encoding block

Figure 2a: Function block of the 32 bit encoding algorithm
Figure 2b: Function block of the 64 bit encoding algorithm
Figure 3: Effect of function f in the encoding algorithm
Figure 4: Data transmission system

DESCRIPTION OF THE PREFERRED EMBODIMENTS

Figure 1 shows the encoding block. With a 64-bit input data format the random number R_{64} which has this format is first converted into a first variant R_8 which consists of 8 bytes: a_7, a_6, a_5, a_4, a_3, a_2, a_1, a_0, where in turn each byte is made up of 8 bits. These 8 bytes are the input data for the encoding algorithm A64 with which a 64 bit random number can be processed. In the application example, the encoding algorithm **A64** is made up of two components, namely **A32A** and **A32B**, where each component processes 32 bits. Of the two components of **A64**, at least one represents an independent encoding algorithm with which a calculation can be performed without consideration of the other part. The **A64** algorithm serves to process a random number with 64-bit format and the **A32A** and **A32B** each serve to process a 32-bit format. When processing the 8 bytes, the first 4 bytes a7, a6, a5, a4 are supplied to the component A32A and the other 4 bytes a3, a2, a1, a0 to A32B. Subsequently, 8 nibbles n_7, n_6, n_5, n_4, n_3, n_2, n_1, n_0 and m_7, m_6, m_5, m_4, m_3, m_2, m_1, m_0 are assigned to the 4 bytes a_7, a_6, a_5, a_4 und a_3, a_2, a_1, a_0 in **A32A** and **A32B** respectively. Each nibble consists of 4 bits. The algorithm and its components include variables. These variables are assigned unique values by means of a 120 bit secret code. This secret code contains the key data which is used for **A64** as well as for **A32A** and **A32B**. They are supplied to the encoding algorithm **A64** from the outside. The encoding algorithm and the secret code must be selected such that they can be used for random numbers or random number variants with different formats. In the application example, this means for a random number or a random number variant with a format of:

64 bits or 16 nibbles or 8 bytes or
32 bits or 8 nibbles or 4 bytes.

Furthermore, the encoding block has a control line with which the format of the random number or the random number variant can be selected by means of a control unit CONTROL. If a 64 bit format is selected via the control line **S64/32**, the encoding algorithm **A64** is activated with its two components **A32A** and **A32B**. The result $\mathbf{E_{32}}$ at the output then has, for example, a 32 bit format. If, however, a 32 bit format is selected for the random number or random number variant via the control line **S64/32**, only the encoding algorithm **A32A** is active. The result $\mathbf{E_{16}}$ at the output then has a 16 bit format. The calculation times are shorter for the smaller format and require less power.

Figure 2a shows the function block of the 32 bit encoding algorithm **A32A**. This is a nibble-oriented encoding which is generated from 32 bits of a random number R_{32} with at least a 32 bit format and which uses, for example, a 120 bit format secret code. By means of this secret code, the nibble permutation σ_A, the function **f**, the first substitution τ_1, the bit permutation σ_B and the second substitution τ_2 can be defined uniquely. Both the two permutations σ_A, σ_B and also the two substitutions can be identical here, so that $\sigma_A = \sigma_B$ and/or $\tau_1 = \tau_2$.
At the beginning eight nibbles $n_7, n_6, n_5, n_4, n_3, n_2, n_1, n_0$, the components of the random number and each consisting of 4 bits, are permutated with σ_A and eight new nibbles $n'_7, n'_6, n'_5, n'_4, n'_3, n'_2, n'_1, n'_0$ are generated. After this, these eight nibbles $n'_7, n'_6, n'_5, n'_4, n'_3, n'_2, n'_1, n'_0$ are supplied to a function **f**. Then the function result is subjected to a first substitution τ_1 after which a further bit permutation σ_B is performed. Finally, a second substitution τ_2 takes place. This result serves to exchange the nibbles $n'_7 \textit{ und } n'_6$ so that a new value with $\bar{n}_7, \bar{n}_6, n'_5, n'_4, n'_3, n'_2, n'_1, n'_0$ is created. This encoding process with the operations described above runs in a loop with, for example, 24 rounds.

Figure 2b shows the function block of the 64 bit encoding algorithm **A64**. This is a byte-oriented coding generated from a 64 bit random number R_{64} and which uses, for instance, a 120 bit format secret code. By means of this secret code, the byte permutation σ_A, the function **f**, the first substitution τ_1, the bit permutation σ_B and the second substitution τ_2 are defined. The two permutations σ_A, σ_B and the two substitutions here can be identical so that $\sigma_A = \sigma_B$ and /or $\tau_1 = \tau_2$.

At the beginning eight bytes $a_7, a_6, a_5, a_4, a_3, a_2, a_1, a_0$, provided by the random number and each consisting of 8 bits, are permutated with σ_A and eight new bytes $a'_7, a'_6, a'_5, a'_4, a'_3, a'_2, a'_1, a'_0$ are generated. After this, these eight bytes $a'_7, a'_6. a'_5. a'_4, a'_3, a'_2, a'_1, a'_0$ are supplied to a function **f**. Then the function result is subjected to a first substitution τ_1 after which a further bit permutation σ_B is performed. Finally, a second substitution τ_2 takes place. This result serves to exchange the byte a'_7 so that a new value with $\bar{a}_7, a'_6, a'_5, a'_4, a'_3, a'_2, a'_1, a'_0$ is created. This encoding process with the operations described above runs in a loop with, for example, 24 rounds.

Figure 3 shows how function **f** operates for the 32 and 64 bit encoding algorithm. Function **f** is structured such that it can work with different input data widths. The input data width can be reduced from 64 bits or 8 bytes $(a_7, a_6, a_5, a_4, a_3, a_2. a_1, a_0)$ or 16 nibbles $(n_7, n_6, n_5, n_4, n_3, n_2. n_1, n_0, m_7, m_6, m_5, m_4, m_3, m_2, m_1, m_0)$ to 32 bits or 4 bytes (a_7, a_6, a_5, a_4) or 8 nibbles $(n_7, n_6, n_5, n_4, n_3, n_2, n_1, n_0)$. In the application example, four bits **hi**, **lo** for example are then assigned to each nibble. Then a logic operation is performed on all four-element bits **hi**, which have originated from the odd-numbered nibbles n_7, n_5, n_3, n_1 $[m_7, m_5, m_3, m_1]$, generating a 4 bit result **hi** irrespective of the number of nibbles. The same happens with the four-element bits **lo**, which have originated from the even-numbered nibbles n_6, n_4, n_2, n_0 $[m_6, m_4, m_2, m_0]$. The end result **20** of the function **f** then always has two nibbles or two four-element bits **hi**, **lo** irrespective of whether the n and m nibbles or the n nibbles alone have generated the 2-nibble result **20**. This Figure is intended to illustrate that in the selection of the 64 bit encoding algorithm both blocks can be activated and in the selection of the 32 bit algorithm only the first block is used, where the same secret code can be used at all times with the same key data for the function **f** of the algorithm. Furthermore, the result that this function **f** supplies from Figures 2a and 2b always has the same format irrespective of the input data format. In this Figure, the control devices **CONTROL**, whose inputs are linked with the even and odd numbered nibble operations and whose output signals generate the result **20**, are influenced by the control line **S64/32**. In this application example, a control line should be understood to mean a line or connection in which, for example, programmable control signals are transferred that influence in software terms the calculation process or the device settings.

Figure 4 shows the data transmission system for variable data formats. The data transmission system has a transponder **1** and a reader **2**. The transponder **1** has a transponder coil **5**, for power and data transmission, and an integrated circuit (IC) **13**. The IC **13** has a transponder control unit **3** for the power supply and sequence control, a memory unit **7** in which the identification number **IDNR** and the secret **CODE** are stored; likewise, intermediate results **ZE** can be stored here and an encoding/decoding/calculation unit **9** for generating the transponder result $\mathbf{E}_{T64/32}$ with the help of the reader's random number $\mathbf{R}_{64}$, $\mathbf{R}_{32}$, which is required as input value for the algorithm **A64**.

The reader **2** consists of a coil **6**, used for both power transmission and data transmission, the control unit 4 for sequence control, the memory unit 8 for storing the identification number **IDNR** and the secret **CODE**, and the encoding/decoding/calculation unit **10** for generating the reader result $\mathbf{E}_{L64/32}$ with the help of the reader's random number $\mathbf{R}_{64}$, $\mathbf{R}_{32}$, which is required as input value for the algorithm **A64**. Furthermore, the arrangement includes an electric, electronic, optical or mechanical switch 12, or alternatively a control line as shown in the preceding Figures with which the format of the random number $\mathbf{R}_{64}$ or $\mathbf{R}_{32}$ is selected.

After activation of the reader **2** - for instance, for a motor vehicle by operating the door handle or by switching on the ignition - power is transmitted from the coil **6** of the reader **2** to the coil 5 of the transponder **1**. This process is shown in the drawing by the arrow marked **POWER** between reader 2 and transponder **1**. The identification number **IDNR**, which is stored in the memory unit **7** of transponder 1, is then sent via the control unit **3** to the reader **2**. Transmission of the identification number is indicated by an arrow with the designation **IDNR** between transponder **1** and reader **2**. The identification number IDNR is verified in the reader 2. A reader random number $\mathbf{R}_{64}/\mathbf{R}_{32}$ is then generated in the reader **2**. The format of the random number $\mathbf{R}_{64}/\mathbf{R}_{32}$ depends on the position of the switch **12**. The random number has either a 64 bit format $\mathbf{R}_{64}$ or a 32 bit format R_{32}. This random number $\mathbf{R_{64}/R_{32}}$ is sent in encoded form to the transponder 1. The random number $\mathbf{R_{64}/R_{32}}$ is decoded in the device **9**. The reader random number $\mathbf{R_{64}/R_{32}}$ and the transponder random number $\mathbf{R_{64}/R_{32}}$ should be identical in the application example. They provide the input data for the calculation with the reader algorithm **A64** and similarly with the transponder algorithm **A64**. In the application example, the transponder algorithm and the reader algorithm are identical and with identical secret **CODE** and identical random number $\mathbf{R_{64}/R_{32}}$ as input variable they generate an identical end result $\mathbf{E_{T32/16}}$, $\mathbf{E_{L32/16}}$ with 32 bit format and 16 bit format respectively. To enhance security, intermediate results **ZE** are generated during calculation. The intermediate result is then used as new input value for the algorithm which then repeats the calculations over several rounds with the constantly changing intermediate results until the end result is obtained after, for example, 24 rounds. The transponder result $E_{T32/16}$ is then sent to the reader **2** where it is compared in a comparator **VGL** with the reader result calculated in the reader **2**.

The selection of the input data width, i.e. the selection as to whether a 64 bit random number R_{64} or a 32 bit random number $\mathbf{R_{32}}$ is selected, can take place not only by means of a switch **12**, as shown in this Figure, but also by means of a programmable control line **S64/32** as portrayed in Figures 1 and 3.

By changing over the input data format for the algorithm, the data sets to be transmitted and hence the power requirement too can be reduced and the reaction speed and range can be increased. With such a data transmission system, it is therefore possible for security steps to be programmed `or set by means of a switch thus allowing subsequent adaptation of the specifications of the data transmission system to satisfy particular requirements.

Data transmission systems of this kind with variable input data format for an encoding algorithm can be used not only for transponder systems but for all wireless transmission systems, especially electromagnetic, optical and high-frequency systems.